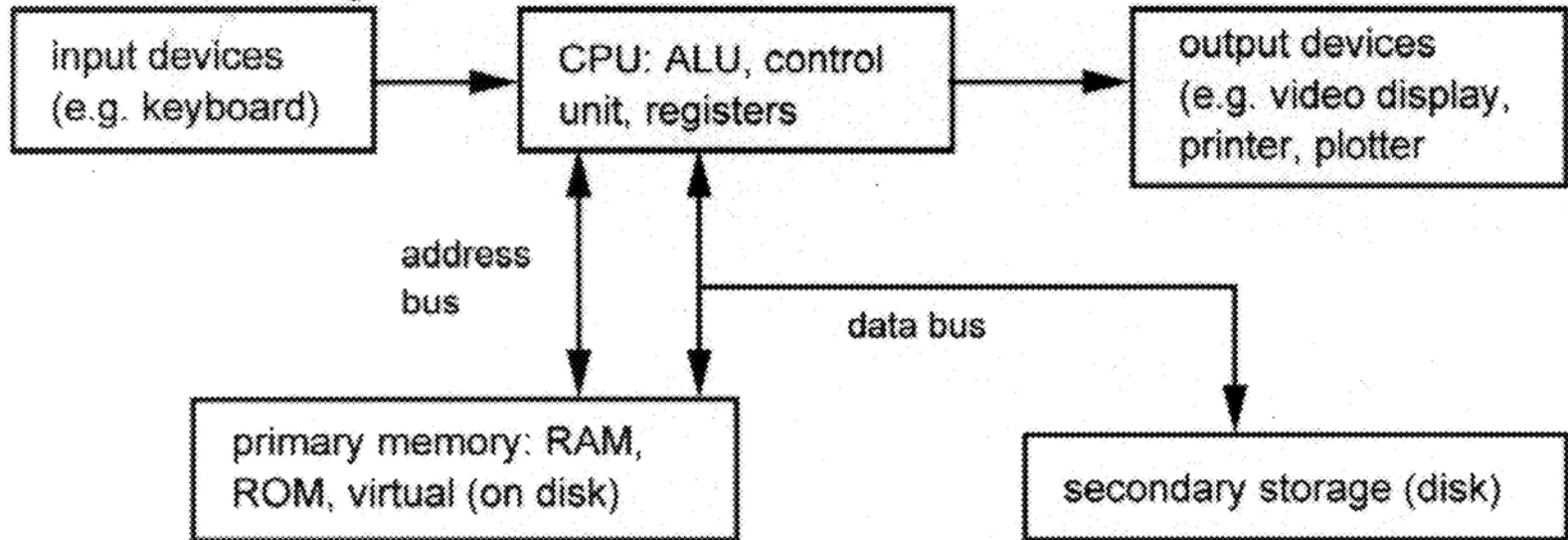


Bio 595

**Computers in
Biomedical Research
Class notes set 2
Fall 2003**

Computer Design

A first look at the organization of a computer.



Programming Languages

ADA: Department of Defense

APL: mathematical language, calculator-like

**BASIC: Beginner's All-purpose Symbolic
Instruction Code; Microsoft Visual Basic 6.0**

**C, C++: structured, modular languages, object
oriented**

COBOL: COmmon Business Oriented Language

Director: Macromedia: multimedia animations

Stella: modeling, simulation, icon-based

Programming Languages

FORTRAN: FORMula TRANslator, high-level for math, science, engineering

HTML: HyperText Markup Language, codes embedded in text

JAVA: high level platform-independent, OOP language similar to C++

PILOT: Programmed Inquiry, Learning or Teaching; application of CAI

PERL: Practical Extraction and Report Language; to process diverse format text files

Software Categories: what you'll use

Operating system

Interpreter

Compiler

Debugger

Editor

Applications software

Other Software (that you won't use)

Assembler

Loader

Linker

Monitor

Diagnostic software

Disassembler

Utility programs

Library programs

Levels of Language

Machine language: Computer 'understands' binary numbers only; commands not easily understood or remembered; an easier to read equivalent: decimal, hexadecimal or octal equivalents to the binary.

Levels of Language

Assembler language: Uses symbolic equivalents for machine codes, symbolic addresses, variable names. Usually there's a 1:1 correspondence between symbols and machine instructions (each of which may take 1-4 bytes or more).

:

Levels of Language

High level language: More concise; one instruction may equal many machine level or assembler instructions. There are numerous languages for general use. In science, there's FORTRAN, ALGOL; in business, it's COBOL; in general use are BASIC, C, JAVA, ADA; compare structured vs. unstructured languages.

Levels of Language

Problem-oriented language: For specialized applications, e.g., SPSS for statistics, LOGO for primary education; PLATO for secondary, college CAI; these languages are themselves written in high level language.

Levels of Language

Comparison of language levels:

High level (BASIC): $w=x+y-z$

Assembler

LD a,x

ADD a,y

SUB a,z

LD w,a

Levels of Language

Object code: depends on where in memory the (assumed byte) values corresponding to w, x, y and z are located. Just suppose that x is at address A004, y is at A005, z at A006 and w at A007. Then

**LD a,x = 3A04A0 (in hex) or
001110100000010010100000
(in binary)**

**ADD a,y = C605A0
(this is a three byte instruction)**

SUB a,z = D606A0

LD w,a = 3207A0

Levels of computer languages

Machine language: Computer 'understands' binary numbers only; commands not easily understood or remembered; an easier to read equivalent: decimal, hexadecimal or octal equivalents to the binary.

Assembler language: Uses symbolic equivalents for machine codes, symbolic addresses, variable names. Usually there's a 1:1 correspondence between symbols and machine instructions (each of which may take 1 to 4 bytes or more).

Levels of computer languages

High level language: More concise; one instruction may equal many machine level or assembler instructions. There are numerous languages for general use. In science, there's FORTRAN, ALGOL; in business, it's COBOL; in general use are BASIC, C, PASCAL, ADA; compare structured vs. unstructured languages.

Problem-oriented language: For specialized applications, e.g., SPSS for statistics, STELLA for simulations; PILOT and LOGO for primary education; PLATO for secondary, college CAI; these languages are themselves written in high level language.

Software (pieces of VB environment)

Operating system: software package designed to control the basic input/output operations of the computer. The OS handles tasks such as loading and running programs, storing data, and multiprocessing (running more than one program at a time). Examples: Windows XP, UNIX 4.2, Macintosh OS 9.

Interpreter: A language translation program: translates one line of high level language at a time, executes the resulting machine language instructions immediately, then does next line, etc., if code doesn't contain errors (i.e., if it is executable). Example: VB 6.0 at runtime.

Software (pieces of VB environment)

Compiler: A language translation program used to transform high level code into machine-readable code.

Source code is high level, machine-independent, while resultant **object code** is low-level, machine-dependent. There isn't a 1:1 comparison between source, object code. Compare: compile time, run time. Example: VB compiler.

Software (pieces of VB environment)

Editor: Program permits entry, storage of source programs. Program being edited is stored in a working area in memory so lines can be modified, deleted, etc. Programs stored in source format (in ASCII code). Example: VB code module window.

Debugger: Program permits checking, correcting of an object (or interpretable source code) program residing in memory. Typical functions: insert breakpoints (program stops, display register contents, CPU status, etc); restart after breakpoint, display memory contents, search for specified value, etc. example: VB in debug mode.

Input and Output Devices

Typewriter-style keyboard, magnetic tape, digitizer tablet, mouse, joystick, paddle, trackball, bar code reader, floppy disk, hard fixed disk, modem, analog-to-digital converter, microphone, video camera, CD ROM, RW CD, DVD

Teletypewriter, dot matrix printer , ink-jet printer, laser printer, DAT, floppy, hard, or removable cartridge disk; optical, pen plotters, TV display tube, raster type, LCD, plasma, fluorescent displays; flat screen active matrix color LCD displays, loudspeakers

Memory Devices: Primary

Primary storage is randomly addressable: one location is as easily (quickly) reached as any other; read/write times are independent of address value. Includes ferrite core (obsolete), semiconductor memories and registers within the CPU.

Semiconductor Memories

RAM (random access memory), dynamic (requires periodic 'refresh' cycle approx. every few msec, works like a reverb unit) or static (retains information as long as power is on).

ROM (read only memory): programs 'burned in' by manufacturer; holds 'firmware', portions of the operating system; information retained when power is off. Information cannot be changed.

EPROM (erasable programmable read only memory): use UV light to erase, power to special pins to program; user-programmable, retains info when power is off.

EEPROM (electrically erasable programmable read-only memory) Major U.S. manufacturers of these 'chip' devices include Intel, AMD, Motorola, etc.

Memory Devices: Secondary

Secondary storage devices are not randomly addressable: access time depends on value of address (determines recording position on the medium).

Magnetic disk is the most common: Floppy diskette: 5 1/4 inch mini-floppy; micro-floppy, 3-1/2 inch diameter (Mac, PC format), 1.4 MB; "superdisks", 120 MB

Memory Devices: Secondary

Hard disk: solid substrate, closer read/write head, higher bit density; can be fixed or removable, usually single disk (or obsolete "pack"); Hard disk drives, fixed or removable; Magnetic drum (obsolete); Magnetic tape: digital: streamer tapes, digital cassette tapes (obsolete)

CD-ROM: CD-R (recordable, erasable); Optical and magneto-optical disk recorders, read-write, WORM

Storage Capacity of Memories

**Core: 4 to 32 K; ROM: 1 K to 128 MB chips;
RAM: 16 to 256 MB; Full sized (8") floppy: 256
KB; Mini-floppy (5-1/4): 100 K (SSSD) to 1.2
MByte (DSQD); Micro-floppy: 1.4 MB HD on
Macintosh; Hard disks: originally 5 and 10
MBytes, now to over 100 GBytes; Removable
media: 100, 220, 750 MB (Zip) or 1 or 2 GB
(Jaz); CD-ROM – 760, 800 MB ; DVD - 5 - 8
Gbytes.**